On Computational Power of Jumping Petri Nets(Semigroups, Formal Languages and Computer Systems)

AUTHOR(S):

TIPLEA, Ferucio Laurentiu

数理解析研究所講究録
960巻 1996年 165-177

On Computational Power of Jumping Petri Nets[1]

Ferucio Laurenţiu ŢIPLEA

Faculty of Informatics
"Al. I. Cuza" University, 6600 Iaşi, Romania
E-mail: fltiplea@infoiasi.ro

Abstract

A *Jumping Petri Net* ([18], [12]), $JPTN$ for short, is defined as a classical net which can spontaneously jumps from a marking to another one. In [18] it has been shown that the reachability problem for $JPTN$'s is undecidable, but it is decidable for finite $JPTN$'s ($FJPTN$). In this paper we investigate the computational power of such nets, via the interleaving semantics. Thus, we show that the non-labelled $JPTN$'s have the same computational power as the labelled or λ-labelled $JPTN$'s. When final markings are considered, the power of $JPTN$'s equals the power of Turing machines. Languages generated by $FJPTN$'s can be represented in terms of regular languages and substitutions with classical Petri net languages. This characterization result leads to many important consequences, e.g. the recursiveness (context-sensitiveness, resp.) of languages generated by arbitrarily labelled (labelled, resp.) $FJPTN$'s. A pumping lemma for nonterminal jumping net languages is also established. Finally, some comparisons between families of languages are given, and a connection between $FJPTN$'s and a subclass of inhibitor nets is presented.

1 Introduction and Preliminaries

It is well-known that the behaviour of some distributed systems cannot be adequately modelled by classical Petri nets. Many extensions which increase the computational and expressive power of Petri nets have been thus introduced. One direction has led to the modification of the firing rule of the nets ([2], [3], [5], [6], [7], [8], [12], [16], [17], [18], [19], [20], [21], [22]).

In this paper we investigate the computational power of jumping Petri nets as introduced in [18]. The paper is organized as follows. In the rest of this section we will establish the basic terminology, notations, and results concerning Petri nets in order to give the reader the necessary prerequisites for the understanding of this paper (for details the reader is referred to [1], [9], [11], [13], [14]). In Section 2 we show that non-labelled $JPTN$'s have the same computational power as labelled or λ-labelled $JPTN$'s and, in the case of final markings, their power equals that of Turing machines. Section 3 gives some characterization results for FJN's in terms of regular languages and substitutions with λ-free languages. Then some important consequences are derived and a pumping lemma for nonterminal jumping net languages is established. In Section 4 some comparisons between families of languages are given. The last section presents a connection between $FJPTN$'s and a subclass of inhibitor nets.

[1] a complete final version of this paper will be published elsewhere

The empty set is denoted by $\emptyset$; for a finite set A, $|A|$ denotes the cardinality of A and $\mathcal{P}(A)$ denotes the set of all subsets of A. Given the sets A and B, $A \subseteq B$ ($A \subset B$, resp.) denotes the inclusion (strict inclusion, resp.) of A in B. If $R \subseteq A \times B$ then $dom(R)$ and $cod(R)$ denote the sets $dom(R) = \{a \in A | \exists b \in B : (a,b) \in R\}$ and $cod(R) = \{b \in B | \exists a \in A : (a,b) \in R\}$. The set of integers (nonnegative integers, positive integers, resp.) is denoted by $\mathbf{Z}$ ($\mathbf{N}$, $\mathbf{N}^+$, resp.).

For a (finite) alphabet V, V^* denotes the free monoid generated by V (under the concatenation operation) with the empty word λ. Given a word $w \in V^*$, $|w|$ denotes the length of w, and $alph(w)$ denotes the set of all letters occurring in w. $\mathcal{L}_3$ ($\mathcal{L}_2$, resp.) denotes the family of regular (context-free, resp.) languages and $\mathcal{L}_{3,pref}$ denotes the family of regular prefix languages (i.e., those languages containing all prefixes of its words).

A *(finite) Petri net*, abbreviated PN, is a 4-tuple $\Sigma = (S, T; F, W)$ where S and T are two finite sets, of *places* and *transitions*, such that $S \cap T = \emptyset$ and $S \cup T \neq \emptyset$, $F \subseteq (S \times T) \cup (T \times S)$ is the *flow relation* and $W : (S \times T) \cup (T \times S) \longrightarrow N$ is the *weight function* of Σ verifying $W(x,y) = 0$ iff $(x,y) \notin F$. A function $M : S \longrightarrow N$ is called a *marking* of Σ and it will be sometimes identified with a vector $M \in N^{|S|}$. The *set of all markings* of Σ is denoted by N^S. The relation "$\leq$" and the operations "+" and "$-$" on nonnegative integers are componentwise extended to vectors in N^S.

A *marked PN*, abbreviated mPN, is a 2-tuple $\gamma = (\Sigma, M_0)$, where Σ is a PN and M_0 is a marking of Σ called the *initial marking* of Σ. An *mPN with final markings*, abbreviated $mPNf$, is a 3-tuple $\gamma = (\Sigma, M_0, \mathcal{M})$, where (Σ, M_0) is an mPN and $\mathcal{M}$, called the *set of final markings* of γ, is a finite set of markings of Σ. A *labelled mPN* ($mPNf$, resp.), abbreviated $lmPN$ ($lmPNf$, resp.), is a 3-tuple (4-tuple, resp.) $\gamma = (\Sigma, M_0, l)$ ($\gamma = (\Sigma, M_0, \mathcal{M}, l)$, resp.), where (Σ, M_0) ($(\Sigma, M_0, \mathcal{M})$, resp.) is an mPN ($mPNf$, resp.) and l, called the *labelling function* of γ, is a function from the set of transitions of Σ into an arbitrary alphabet V. A *λ-labelled mPN* ($mPNf$, resp.), abbreviated $l^\lambda mPN$ ($l^\lambda mPNf$, resp.), is a 3-tuple (4-tuple, resp.) $\gamma = (\Sigma, M_0, l)$ ($\gamma = (\Sigma, M_0, \mathcal{M}, l)$, resp.), where (Σ, M_0) ($(\Sigma, M_0, \mathcal{M})$, resp.) is an mPN ($mPNf$, resp.) and l, called the *λ-labelling function* of γ, is a function from the set of transitions of Σ into $V \cup \{\lambda\}$, where V is an arbitrary alphabet and λ is the unity of V^*.

In the sequel we shall often use the term "net" whenever we refer to a structure γ as those above. The notations ${}^\bullet x = \{(y,x) \mid y \in S \cup T, (y,x) \in F\}$ and $x^\bullet = \{(x,y) \mid y \in S \cup T, (x,y) \in F\}$, for all $x \in S \cup T$, will be intensively used. Pictorially, a net γ will be represented by a graph. Then places are denoted by circles, transitions are denoted by boxes; the flow relation is represented by direct edges: there is a direct edge from x to y iff $(x,y) \in F$. The initial marking is given by putting $M_0(s)$ tokens in the circle representing the place s. The labelling function is denoted by placing letters into the boxes representing transitions and final markings are explicitly listed.

Let γ be a net and $M \in N^S$. A transition $t \in T$ is *enabled* at M, denoted $M[t\rangle$, if $M(s) \geq W(s,t)$ for all $s \in S$. If t is enabled at M then t may *occur* yielding a new marking M', abbreviated $M[t\rangle M'$, given by $M'(s) = M(s) - W(s,t) + W(t,s)$ for all $s \in S$. The above definition can be naturally extended to sequences of transitions by: $M[\lambda\rangle M$ and $M[ut\rangle M'$ iff there is a marking M'' such that $M[u\rangle M''[t\rangle M'$ (M and M' are markings, $u \in T^*$ and $t \in T$). Moreover, if $M_0[w\rangle M$ then we say that w is a *firing* or *transition*

sequence of γ (leading from M_0 to M), and M ia *reachable* (from M_0) in γ. The set of all reachable markings of γ is denoted by $[M_0\rangle$.

Petri nets can be viewed as language generators. First, we extend the labelling function to morphisms in the usual way. Then, let γ_1 be an $mPTN$, γ_2 either an $lmPTN$ or an $l^\lambda mPTN$, γ_3 an $mPTNf$, and γ_4 either an $lmPTNf$ or an $l^\lambda mPTNf$. The languages generated by these nets are $P(\gamma_1) = \{w | w \in T^* \wedge (\exists M \in \mathbf{N}^S : M_0[w\rangle_{\gamma_1} M)\}$, $P(\gamma_2) = \{l(w) | w \in T^* \wedge (\exists M \in \mathbf{N}^S : M_0[w\rangle_{\gamma_2} M)\}$, $L(\gamma_3) = \{w | w \in T^* \wedge (\exists M \in \mathcal{M} : M_0[w\rangle_{\gamma_3} M)\}$, $L(\gamma_4) = \{l(w) | w \in T^* \wedge (\exists M \in \mathcal{M} : M_0[w\rangle_{\gamma_4} M)\}$. The languages generated by $mPTN$ ($lmPTN$, $l^\lambda mPTN$, resp.) are called *free P-type languages* (*P-type languages*, *arbitrary P-type languages*, resp.) and the family of these languages is denoted by $\mathbf{P^f}$ ($\mathbf{P}$, $\mathbf{P}^\lambda$, resp.). For Petri nets with final markings, the terminology is as above, by changing "P" into "L". These languages are usually referred to as *Petri net languages* or *Petri languages*.

A *jumping P/T-net* ([18]), abbreviated $JPTN$, is a pair $\gamma = (\Sigma, R)$, where Σ is a PTN and R, the *set of spontaneous jumps* of γ, is a binary relation on the set of markings of Σ. In what follows the set R of spontaneous jumps of $JPTN$'s will be assumed *recursive*, that is for any couple (M, M') we can effectively decide whether or not (M, M') is a member of R. If γ has finitely many jumps then we say that γ is a *finite* $JPTN$, abbreviated $FJPTN$.

Let $Y \in \{JPTN, FJPTN\}$. As usual one can define an mY (mYf, lmY, $lmYf$, $l^\lambda mY$, $l^\lambda mYf$, resp.) by adding an initial marking (a set of final markings, a labelling, a λ-labelling, resp.) to Y. In fact, all remarks about Petri nets equally hold for jumping Petri nets. Pictorially, a jumping net will be represented as a classical net. Moreover, the relation R will be separately listed.

Let γ be a jumping net. The transition t is *j-enabled at a marking* M (in γ), abbreviated $M[t\rangle_{\gamma,j}$, iff there exists a marking M_1 such that $MR^*M_1[t\rangle_\Sigma$ (Σ being the underlying net of γ and R^* the reflexive and transitive closure of R); if t may occur at M then it can yield a new marking M', abbreviated $M[t\rangle_{\gamma,j}M'$, given by $MR^*M_1[t\rangle_\Sigma M_2R^*M'$, where M_1, M_2 are markings of γ. The notions of *transition j-sequence* and *j-reachable marking* are similarly defined as for Petri nets (we set $M[\lambda\rangle_{\gamma,j}M'$ whenever MR^*M'). The *set of all j-reachable markings* of a marked JN γ is denoted by $[M_0\rangle_{\gamma,j}$ (M_0 being the initial marking of γ). The notation "$[\cdot\rangle_{\gamma,j}$" will be simplified to "$[\cdot\rangle_j$" whenever γ is understood from the context.

Jumping nets can be considered as generators of languages in the same way as classical nets, by changing "$[\cdot\rangle$" into "$[\cdot\rangle_j$". For example, if $\gamma = (\Sigma, R, M_0, \mathcal{M}, l)$ is an $l^\lambda mJPTNf$, then the language generated by γ is $L(\gamma) = \{l(w) | w \in T^* \wedge (\exists M \in \mathcal{M} : M_0[w\rangle_j M)\}$. Thus, $\mathbf{RX^f}$ ($\mathbf{RX}$, $\mathbf{RX}^\lambda$, resp.) will denote the family of *free X-type jumping Petri net languages* (*X-type jumping Petri net languages*, *arbitrary X-type jumping Petri net languages*, resp.), for any $X \in \{P, L\}$. For finite jumping nets, the corresponding family of languages will be denoted by $\mathbf{RX^f_{fin}}$ ($\mathbf{RX_{fin}}$, $\mathbf{RX^\lambda_{fin}}$, resp.). For any $X \in \{P, L\}$ we have $\mathbf{X^f} \subseteq \mathbf{RX^f_{fin}} \subseteq \mathbf{RX^f}$, $\mathbf{X} \subseteq \mathbf{RX_{fin}} \subseteq \mathbf{RX}$, $\mathbf{X}^\lambda \subseteq \mathbf{RX^\lambda_{fin}} \subseteq \mathbf{RX}^\lambda$.

Some jumps of a FJN can be never used. Thus we say that a marked finite jumping net γ is *R-reduced* if for any jump (M, M') of γ we have $M \neq M'$, $M \in [M_0\rangle_{\gamma,j}$, and there are some final markings of γ which are j-enabled from M' (if γ has final markings). As

the reachability problem for FJN's is decidable ([18]), for any marked FJN γ we can effectively construct (modifying only the set of jumpings of γ) a marked FJN γ' such that γ' is R-reduced and it has the same computational power as γ. All finite jumping nets in this paper will be considered R-reduced.

2 Jumps and Labellings

In this section we show that the jumps can "simulate" the labelling of nets. Then we use this result to prove that the power of $JPTN$'s equals the class of recursively enumerable languages. In the case of jumping nets with finite state space the connection with regular languages is made.

Theorem 2.1 *For any* $X \in \{P, L\}$ *we have* $\mathbf{RX^f} = \mathbf{RX} = \mathbf{RX}^\lambda$.

Sketch of Proof We prove only the case $X = L$, the other one being similar to this. The inclusion $\mathbf{RL^f} \subseteq \mathbf{RL}$ follows from definitions. Conversely, let $L \in \mathbf{RL}$. There is an $lmJPTNf$ $\gamma = (\Sigma, R, M_0, \mathcal{M}, l)$ such that $L = L(\gamma)$. Let $\Sigma = (S, T; F, W)$. Without loss of the generality we may assume $T \cap \{l(t) | t \in T\} = \emptyset$. Let $T_1 = \{t \in T | \forall t' \in T,\ t \neq t' \Longrightarrow l(t) \neq l(t')\} \subseteq T$.

If $T_1 = T$ we consider $\gamma' = (\Sigma', R, M_0, \mathcal{M})$, where Σ' is obtained from Σ renaming each transition t by $l(t)$. γ' is an $mJPTNf$ and $L(\gamma') = L$. If $T_1 \subset T$, we construct $\gamma' = (\Sigma, R', M_0', \mathcal{M}')$ as described bellow. We partition the set $T_2 = T - T_1$ in $k \geq 1$ subsets, $T_2 = T_2^1 \cup \cdots \cup T_2^k$, such that for any i, $1 \leq i \leq k$, the set T_2^i contains those transitions of Σ which have the same label; let a_i be this label. We have $a_i \neq a_j$ for any $i \neq j$. The set of transitions of Σ' will be $T' = l(T_1) \cup T_2 \cup \{a_1, \ldots, a_k\}$. The basic idea is: when a transition $t \in T_1$ occurs in γ, its effect is simulated in γ' by the transition $l(t) \in l(T_1)$; when a transition $t \in T_2^i$, $1 \leq i \leq k$, occurs in γ, its effect is simulated in γ' by the relation R' and the transition a_i. The transitions of T_2 will be blocked forever in the net γ'.

We consider now the second equality. The inclusion $\mathbf{RL} \subseteq \mathbf{RL}^\lambda$ follows from definitions. Conversely, let $L \in \mathbf{RL}^\lambda$. There is an $l^\lambda mJPTNf$ $\gamma = (\Sigma, R, M_0, \mathcal{M}, l)$ such that $L = L(\gamma)$. Without loss of the generality we may assume that $T \cap \{l(t) | t \in T\} = \emptyset$, T being the set of transitions of γ. Let $T_1 = \{t \in T | l(t) \neq \lambda\}$. We have $T_1 \subseteq T$.

If $T_1 = T$ then γ is an $lmJPTNf$ and hence $L \in \mathbf{RL}$. If $T_1 \subset T$ we construct an $lmJPTNf$ $\gamma' = (\Sigma', R', M_0', \mathcal{M}', l')$ as follows. Σ' will have the same transitions and places as Σ excepting a new place s' which will be used to block all transitions in $T_2 = T - T_1$; their effect will be simulated by the relation R'. □

Theorem 2.2 $\mathbf{RL^f} = \mathbf{RL} = \mathbf{RL}^\lambda = \mathcal{L}_0$.

Proof The equalities $\mathbf{RL^f} = \mathbf{RL} = \mathbf{RL}^\lambda$ have been already established. The equality with the set of all recursively enumarable languages can be obtained as follows. In [18] it has been proved that jumping Petri nets can simulate inhibitor nets (which have the power of Turing machines). As a consequence, $\mathcal{L}_0 \subseteq \mathbf{RL}^\lambda$. Now we prove that $\mathbf{RL^f} \subseteq \mathcal{L}_0$. Let $\gamma = (\Sigma, R, M_0, \mathcal{M})$ be an $mJPTNf$. We show that there is an algorithm $\mathcal{A}$ such that for all $w \in T^*$ we have:

$w \in L(\gamma)$ iff $\mathcal{A}$ beginning with the input w it will eventually halt accepting w.

First we have to remark that $w \in L(\gamma)$ iff there is a computation in γ of the form:

$$M_0R^*M_0'[w_1\rangle M_1R^+M_1' \cdots M_{k-1}R^+M_{k-1}'[w_k\rangle M_kR^*M \in \mathcal{M},$$

where $w_1, \ldots, w_k$ $(k \geq 1)$ is a decomposition of w in non-empty words, that is $w = w_1 \cdots w_k$ and none of w_i is empty. All computations as the above one will be called *terminal computations* in γ. A terminal computation can be written as a (formal) string

$$(M_0, M_0')w_1(M_1, M_1') \cdots (M_{k-1}, M_{k-1}')w_k(M_k, M),$$

where $(M_0, M_0'), (M_k, M) \in R^*$, $(M_1, M_1'), \ldots, (M_{k-1}, M_{k-1}') \in R^+$ and $w_1, \ldots, w_k \in T^+$ (the empty transition sequence is identified by a string of the form $(M_0R^*M_0')$). It is clear that not any string as above describes a terminal computation in γ. But if we have such a string we can effectively decide whether or not it describes a terminal computation in γ. Now we remark that R^* is recursively enumerable (R is recursive) and consequently, we can enumerate R^*, $r_0, r_1, \ldots, r_n, \ldots$, (for any $n \geq 0$, r_n is a couple (M, M') satisfying MR^*M').

Any $w \in T^*$ has finitely many decompositions $w = w_1 \cdots w_k$ $(k \geq 1)$ with $w_i \in T^+$ for all i, and let $d_1, \ldots, d_m$ $(m \geq 1)$ be all these decompositions. For any decomposition d_i $(1 \leq i \leq m)$, $d_i : \quad w = w_1 \cdots w_{k_i}$, we consider the N-indexed sequence S_i defined by:

- consider first all strings obtained from d_i and r_0 as above (in this case we have only one string $r_0w_1r_0 \cdots r_0w_{k_i}r_0$);
- consider then, in an arbitrary but fixed order, all strings as above obtained from d_i and r_0, r_1 (for example, $r_0w_1r_0 \cdots r_0w_{k_i}r_1$ is such a string);
- and so on.

We obtain, using all decompositions of w, m sequences $S_i : \ c_1^m, c_2^m, \ldots c_n^m, \ldots$, $1 \leq i \leq m$. Now, the activity of the algorithm $\mathcal{A}$ on the input $w \in T^*$ can be described as follows:

1. $\mathcal{A}$ computes all decompositions of w; let $d_1, \ldots, d_m$ $(m \geq 1)$ be these decompositions;
2. $\mathcal{A}$ searchs the sequences $S_1, \ldots, S_m$ (as above) in the order
$$c_1^1, c_1^2, \ldots, c_1^m, c_2^1, c_2^2, \ldots, c_2^m, \ldots$$
3. for a string c_i^j $(i \geq 1, 1 \leq j \leq m)$ the algorithm $\mathcal{A}$ can effectively decide whether or not c_i^j describes a terminal computation of w in γ. If this is the case, then $\mathcal{A}$ halts with the answer "w is a member of $L(\gamma)$"; otherwise, $\mathcal{A}$ will continue the searching.

It is easy to see that $\mathcal{A}$ halts on the input w iff $w \in L(\gamma)$. We conclude that $L(\gamma) \in \mathcal{L}_0$ and so, $\mathbf{RL^f} \subseteq \mathcal{L}_0$. Combining this inclusion with the other one we obtain the theorem. □

3 Characterization Results and Consequences

In this section we focus on finite jumping nets. We shall prove that any language $L \in \mathbf{RL^f_{fin}}$ ($\mathbf{RL_{fin}}$, $\mathbf{RL^\lambda_{fin}}$, resp.) can be represented as $L = \varphi(L')$, where L' is a regular language and φ is a substitution with λ-free languages. Similar results hold true for P-type jumping Petri net languages.

Theorem 3.1 *For any $L \in \mathbf{RL^f_{fin}}$ ($\mathbf{RL_{fin}}$, $\mathbf{RL^\lambda_{fin}}$, resp.) there exist a language $L' \in \mathcal{L}_3$ and a substitution with λ-free languages φ from $alph(L')$ into $\mathbf{L^f}$ ($\mathbf{L}$, $\mathbf{L^\lambda}$, resp.) such that $L = \varphi(L')$.*

Proof Let $L \in \mathbf{RL^f_{fin}}$. There is an $mFJPTNf$ $\gamma = (\Sigma, R, M_0, \mathcal{M})$ such that $L = L(\gamma)$. We construct an finite automaton with λ-moves, $A = (Q, I, \delta, q_0, Q_f)$, as follows:

(i) $Q = \{M_0\} \cup dom(R) \cup cod(R) \cup \mathcal{M}$, $q_0 = M_0$, $Q_f = \mathcal{M}$;
(ii) $I = \{a_{M',M} | M', M \in Q$ and M is reachable from M' in Σ by a non-empty sequence of transitions$\}$;
(iii) $\delta : Q \times (I \cup \{\lambda\}) \longrightarrow Q$ is given by: $\delta(M', a_{M',M}) = \{M\}$ if $a_{M,M'} \in I$, $\delta(M, \lambda) = \{M' | (M, M') \in R\}$; it is undefined otherwise.

Let $L' = L(A)$ and $\varphi : alph(L') \longrightarrow \mathbf{L^f}$ given by $\varphi(a_{M',M}) = L(M', M) - \{\lambda\}$, where $L(M', M)$ is the language generated by the $mPTNf\ (\Sigma, M', \{M\})$.

We have $L' \in \mathcal{L}_3$. Let us prove that $L = \varphi(L')$. First, $\lambda \in L$ iff $\lambda \in L'$ and hence $\lambda \in L$ iff $\lambda \in \varphi(L')$. Let now $w \in L$, $w \neq \lambda$. There is a decomposition of w, $w = w_1 \cdots w_{m+1}$, $m \geq 0$, such that $M_0 R^* M_0'[w_1\rangle_\Sigma M_1 R^+ M_1' \ldots R^+ M_m'[w_{m+1}\rangle_\Sigma M_{m+1} R^* M_{m+1}' \in \mathcal{M}$, where M_i and M_i' are markings of γ and $w_i \neq \lambda$ for any $0 \leq i \leq m+1$.

The sequence $u = a_{M_0',M_1} a_{M_1',M_2} \cdots a_{M_m',M_{m+1}}$ determines a unique path, excepting λ-moves, from M_0 to M_{m+1}' in the automaton A; hence $u \in L'$. For any i, $1 \leq i \leq m+1$, we have $w_i \in L(M_{i-1}', M_i) - \{\lambda\}$ which shows that $w \in \varphi(u)$, i.e. $w \in \varphi(L')$. Thus the inclusion $L \subseteq \varphi(L')$ is proved. The other inclusion can be analogously proved.

The case $L \in \mathbf{RL_{fin}}$ can be simply settled starting from the remark that if $L = L(\gamma)$, $\gamma = (\Sigma, R, M_0, \mathcal{M}, l)$, then $L = l(L(\gamma'))$, where $\gamma' = (\Sigma, R, M_0, \mathcal{M})$. Now, there exist a regular language L' and a substitution with λ-free languages ψ from $alph(L')$ into $\mathbf{L^f}$ such that $L(\gamma') = \psi(L')$. Define $\varphi = l \circ \psi$ which is a substitution with λ-free languages, and obtain $L = \varphi(L')$.

The previous idea does not work for the family $\mathbf{RL_{fin}^\lambda}$ because l is an arbitrary labelling function and, for some a, $(l \circ \psi)(a)$ could contain λ. But we will modify the construction in the case of $\mathbf{RL_{fin}^f}$ by setting $\varphi(a_{M',M}) = l(L(M', M)) - \{\lambda\}$, for any $a_{M',M}$, and adding to the automaton A the arcs $(\overline{M}, \overline{M'})$ labelled by λ whenever there exist in A the arcs $(\overline{M}, M')$ and $(M, \overline{M'})$ labelled by λ and (M', M) labelled by $a_{M',M}$ and $\lambda \in l(L(\Sigma, M', \{M\}))$ (we mention that Petri net languages are recursive languages ([11]) and so we can effectively decide whether or not λ is a member of such language). It is easy to see that the theorem holds also true in this case. □

The proof of Theorem 3.1 is effective. This fact permits us to show that terminal jumping Petri net languages are recursive.

Corollary 3.1 $\mathbf{RL_{fin}^\lambda} \subseteq \mathcal{L}_{rec}$.

Proof We show that the membership problem for the family $\mathbf{RL_{fin}^\lambda}$ is decidable. Let $\gamma = (\Sigma, R, M_0, \mathcal{M}, l)$ be an $l^\lambda mFJPTNf$, T the set of its transitions and V the range of l. From Theorem 3.1 it follows that we can effectively compute a regular language L' (given by a finite automaton) and a substitution with λ-free languages $\varphi : alph(L') \longrightarrow \mathbf{L}^\lambda$ such that $L(\gamma) = \varphi(L')$. Let $w \in V^*$. Since φ is a substitution with λ-free languages we have:

- if $w = \lambda$ then $w \in L(\gamma)$ iff $w \in L'$;
- if $w \neq \lambda$, $w = a_1 \cdots a_n$ $(n \geq 1)$, then $w \in L(\gamma)$ iff there exist $w = b_1 \cdots b_m \in L'$ $(1 \leq m \leq n)$ and $u_i \in \varphi(b_i)$, $1 \leq i \leq n$, such that $|u_i| \leq n$ and $w = u_1 \cdots u_m$.

Consequently, the membership problem for $L(\gamma)$ can be reduced to the membership problem for a regular language and for some arbitrary Petri net languages. As the arbitrary Petri net languages are recursive ([11]) we conclude that the membership problem for $\mathbf{RL_{fin}^\lambda}$ is decidable. □

Corollary 3.2 $\mathbf{RL_{fin}} \subseteq \mathcal{L}_1$.

Proof For any language $L \in \mathbf{RL_{fin}}$ there exist a regular language L and a substitution with λ-free languages φ from $alph(L')$ into $\mathbf{L}$ such that $L = \varphi(L')$. But $\mathbf{L} \subset \mathcal{L}_1$ ([11]) and $\mathcal{L}_1$ is closed under substitutions with λ-free languages, from which the theorem follows. □

The converse of Theorem 3.1 holds true for labelled or λ-labelled jumping nets.

Theorem 3.2 *If $L \in \mathcal{L}_3$ and φ is a substitution from $alph(L)$ into $\mathbf{L}$ ($\mathbf{L}^\lambda$, resp.) then $\varphi(L) \in \mathbf{RL_{fin}}$ ($\mathbf{RL^\lambda_{fin}}$, resp.).*

Sketch of Proof Let $L \in \mathcal{L}_3$ and $\varphi : alph(L) \longrightarrow \mathbf{L}$. There is an $lmPTNf$ $\gamma = (\Sigma, M_0, \mathcal{M}, l)$ such that $L(\gamma) = L$. Moreover, $[M_0\rangle$ is finite. Let $alph(L) = \{a_1, \ldots, a_n\}$, $n \geq 1$, and $L_i = \varphi(a_i)$, $1 \leq i \leq n$. There exist the $lmPTNf$ $\gamma_i = (\Sigma_i, M^i_0, \mathcal{M}_i, l_i)$, $1 \leq i \leq n$, such that $L_i = L(\gamma_i)$.

Construct an $lmJPTNf$ $\gamma' = (\Sigma', R', M'_0, \mathcal{M}', l')$ such that $\varphi(L) = L(\gamma')$, starting from the following idea. The nets $\Sigma, \Sigma_1, \ldots, \Sigma_n$ will be subnets of Σ' and initially they will be "blocked". When a transition t labelled by $l(t) = a_i$, $1 \leq i \leq n$, occurs in γ then in γ' the subnet Σ_i will be relieved (by means R') and a transition sequence w in γ_i can now occur in γ'. When a final marking will be reached in Σ_i this subnet will be blocked again by means of R'. □

Corollary 3.3 *$L \in \mathbf{RL_{fin}}$ ($\mathbf{RL^\lambda_{fin}}$, resp.) iff there exist $L' \in \mathcal{L}_3$ and a substitution with λ-free languages $\varphi : alph(L) \to \mathbf{L}$ ($\mathbf{L}^\lambda$, resp.) such that $L = \varphi(L')$.*

A similar result as that in Theorem 3.1 holds for P-type jumping languages.

Theorem 3.3 *For any $L \in \mathbf{RP^f_{fin}}$ ($\mathbf{RP_{fin}}$, $\mathbf{RP^\lambda_{fin}}$, resp.) there exist a language $L' \in \mathcal{L}_{3,pref}$, a substitution with λ-free languages φ from $alph(L')$ into $\mathbf{L^f}$ ($\mathbf{L}$, $\mathbf{L}^\lambda$, resp.), and the languages P_0 and P_a, $a \in alph(L')$, such that*

$$L = P_0 \cup \bigcup_{a \in alph(L')} \varphi(\partial^r_a(L')\{a\})P_a$$

(∂^r denotes the right derivative). Moreover, the languages P_0 and P_a, $a \in alph(L')$, are finite unions of free P-type languages (P-type languages, arbitrary P-type languages, resp.).

Proof Let $\gamma = (\Sigma, R, M_0)$ be an $mFJPTN$ such that $L = L(\gamma)$. We construct an finite automaton with λ-moves, $A = (Q, I, \delta, q_0, Q_f)$, similar to that described in the proof of Theorem 4.1, excepting only the sets of states anf final states which will be $Q = \{M_0\} \cup dom(R) \cup cod(R)$ and $Q_f = Q$. Next we consider $L' = L(A)$ which is a prefix regular language, the substitution φ as in the proof of Theorem 3.1, $P_0 = \bigcup_{(M_0, M) \in R^*} P(\Sigma, M)$, and $P_{a_{M',M}} = \bigcup_{(M,M'') \in R^+} P(\Sigma, M'')$ for any $a_{M',M} \in alph(L')$. Now, let us prove the equality in theorem. Let $w \in L$.

If $w = \lambda$ or the computation induced by w contains a group of jumps only at the beginning ($M_0 R^* M'_0[w\rangle M$) then $w \in P_0$. Otherwise there is a decomposition of w, $w = w_1 \cdots w_{m+1}$, $m \geq 1$, such that $M_0 R^* M'_0[w_1\rangle M_1 R^+ M'_1 \ldots [w_m\rangle M_m R^+ M'_m[w_{m+1}\rangle M \in \mathbf{N}^S$, where $w_i \neq \lambda$ for any $1 \leq i \leq m+1$.

The sequence $u = a_{M'_0,M_1} a_{M'_1,M_2} \ldots a_{M'_{m-1},M_m}$ determines a unique path (from M_0 to M_m) in the automaton A and hence $u \in L'$. For any i, $1 \leq i \leq m$, we have $w_i \in L(M'_{i-1}, M_i) = \varphi(a_{M'_{i-1},M_i})$ which shows that $w_1 \cdots w_m \in \varphi(u)$, i.e. $w_1 \cdots w_m \in \varphi(u) \subseteq$

$\varphi(\partial^r_{a(M'_{m-1},M_m)}(L')\{a_{M'_{m-1},M_m}\})$. But, it is clear that $w_{m+1} \in P_{a_{M'_{m-1},M_m}}$, and thus we obtain

$$w \in \varphi(\partial^r_{a_{M'_{m-1},M_m}}(L')\{a_{M'_{m-1},M_m}\})P_{a_{M'_{m-1},M_m}} \subseteq \bigcup_{a\in alph(L')} \varphi(\partial^r_a(L')\{a\})P_a.$$

The other inclusion can be analogously proved.

The case $L \in \mathbf{RL_{fin}}$ ($L \in \mathbf{RL^{\lambda}_{fin}}$, resp.) can be settled as in the proof of Theorem 4.1. We only mention that the languages P_0 and P_a are images by the labelling homomorphism l of finite unions of free P-type Petri net languages; that is, P_0 and P_a are finite unions of P-type Petri net languages (arbitrary P-type Petri net languages, resp.). □

Corollary 3.4 *For any $L \in \mathbf{RP_{fin}}$ ($\mathbf{RP^{\lambda}_{fin}}$, resp.) there exist a language $L' \in \mathcal{L}_{3,pref}$, a substitution with λ-free languages φ from $alph(L')$ into $\mathbf{L}$ ($\mathbf{L}^{\lambda}$, resp.), and the P-type languages (arbitrary P-type languages, resp.) P_0 and P_a, $a \in alph(L')$, such that*

$$L = P_0 \cup \bigcup_{a\in alph(L')} \varphi(\partial^r_a(L')\{a\})P_a.$$

Proof The family of (arbitrary) P-type languages is closed under union ([11]). □

The idea of proof of Theorem 4.2 cannot be used for the family $\mathbf{RL^f_{fin}}$ because it is not generally true that $T_i \cap T_j = \phi$ for any $i \neq j$, and it cannot be used for P-type languages because the relation R^2 is, in general, infinite.

Using similar constructions as for classical Petri net languages it is easy to prove that the families $\mathbf{RL_{fin}}$ and $\mathbf{RL^{\lambda}_{fin}}$ are closed under finite union and catenation (one can use also the power of jumping relation in corelation with final markings). Then we have:

Corollary 3.5 $\mathbf{RP_{fin}} \subseteq \mathbf{RL_{fin}}$ *and* $\mathbf{RP^{\lambda}_{fin}} \subseteq \mathbf{RL^{\lambda}_{fin}}$.

Proof We will prove only the inclusion $\mathbf{RP_{fin}} \subseteq \mathbf{RL_{fin}}$, the other one being similar to this one. If $L \in \mathbf{RP_{fin}}$ then L can be written as in Theorem 3.3:

$$L = P_0 \cup \bigcup_{a\in alph(L')} \varphi(\partial^r_a(L')\{a\})P_a.$$

$\partial^r_a(L')\{a\}$ is a regular language and so $\varphi(\partial^r_a(L')\{a\}) \in \mathbf{RL_{fin}}$ for any $a \in alph(L')$ (Theorem 3.2).

It is well-known that P-type Petri net languages are also L-type Petri net languages ([11]), that is $\mathbf{P} \subseteq \mathbf{L}$, and so $P_0, P_a \in \mathbf{L} \subseteq \mathbf{RL_{fin}}$. Using the above remark, that is $\mathbf{RL_{fin}}$ is closed under finite union and catenation, we obtain $L \in \mathbf{RL_{fin}}$. □

For P-type languages the next kind of pumping lemma holds true.

Theorem 3.4 *For any $L \in \mathbf{RP^{\lambda}_{fin}}$ there is a number $k \in \mathbf{N}$ such that for each word $w \in L$, if $|w| \geq k$ then there is a prefix w' of w which has a decomposition $w' = xyz$ such that $|y| \geq 1$ and $xy^{m+1}z \in L$ for all $m \geq 0$.*

Proof Let $\gamma = (\Sigma, R, M_0, l)$ be an $l^{\lambda}mFJPTN$ such that $L = P(\gamma)$. Consider the automaton A, the substitution φ and the languages L', P_0 and P_a ($a \in I$) as in the proof of Theorem 4.3 (the languages P_0 and P_a, $a \in I$, are arbitrary P-type Petri net languages). We have:

$$L = P_0 \cup \bigcup_{a\in alph(L')} \varphi(\partial^r_a(L')\{a\})P_a.$$

Let k_1, k_0, k_a ($a \in I$) be the constants from the pumping lemmata for the regular language L' ([10]) and for the arbitrary P-type Petri net languages P_0 and P_a, $a \in I$ ([4]). Consider $k_2 = max\{k_0, k_a | a \in I\}$ and $k = k_1k_2$. We shall prove that the number k satisfies the theorem.

Let $w \in L$ such that $|w| \geq k$. If $w \in P_0$ then we apply the pumping lemma for w with respect to P_0 and we obtain the theorem, with $w' = w$. Otherwise, there is a word $u = a_1 \cdots a_s \in L'$ such that $w \in \varphi(u)P_{a_s}$. We have to consider two cases.

Case 1 $s \geq k_1$. From the pumping lemma for regular languages, u has a decomposition $u = u_1u_2u_3$ such that $|u_2| \geq 1$ and $u_1u_2^iu_3 \in L'$ for any $i \geq 0$. Since $w \in \varphi(u)P_{a_s} = \varphi(u_1)\varphi(u_2)\varphi(u_3)P_{a_s}$, there exist $x \in \varphi(u_1)$, $y \in \varphi(u_2)$, $z \in \varphi(u_3)$ and $v \in P_{a_s}$ such that $w = xyzv$. φ being a substitution with λ-free languages it follows that $|y| \geq 1$.

From $u_1u_2^iu_3 \in L'$ it follows that $\varphi(u_1)[\varphi(u_2)]^i\varphi(u_3)P_{a_s} \subseteq L$ for any $i \geq 0$. Hence, $xy^izv \in L$ for any $i \geq 0$, and the theorem is satisfied with $w' = w$.

Case 2 $s < k_1$. From $w \in \varphi(a_1 \cdots a_s)P_{a_s}$ it follows that there exist $w_j \in \varphi(a_j)$, $1 \leq j \leq s$, and $w_{s+1} \in P_{a_s}$ such that $w = w_1 \cdots w_sw_{s+1}$. Since $|w| \geq k = k_1k_2$ and $|w| = |w_1| + \ldots + |w_s| + |w_{s+1}|$ and $s < k_1$, there is $j \in \{1, \ldots, s+1\}$ such that $|w_j| > k_2 \geq k_{a_j}$.

If $j = s+1$ then we apply the pumping lemma for the language P_{a_s} and we obtain the theorem with $w' = w$.

If $j = 1$ then it is easy to remark that $L(M_0', M_1) \subseteq P_0$, where $a_1 = a_{M_0', M_1}$ and $M_0R^*M_0'[w_1 > M_1$. Thus $w_1 \in P_0$, and now we have to apply the pumping lemma for the word w_1 with respect to P_0. Then $w_1 = x_1y_1z_1$ with $|y_1| \geq 1$ and $x_1y_1^iz_1 \in P_0$ for any $i \geq 1$. Consider $w' = w_1$, $x = x_1$, $y = y_1$ and $z = z_1$ and the theorem is satisfied.

If $1 < j < s+1$ then let us suppose that $a_{j-1} = a_{M_{j-2}', M_{j-1}}$ and $a_j = a_{M_{j-1}', M_j}$. Then, $\varphi(a_j) = L(M_{j-1}', M_j) = L(\Sigma, M_{j-1}', \{M_j\})$ and $P_{a_{j-1}} = \bigcup_{(M_{j-1}, M) \in R^+} P(\Sigma, M)$. Since $M_{j-1}R^+M_{j-1}'$ it follws that $\varphi(a_j) \subseteq P_{a_{j-1}}$ and $\varphi(a_1) \cdots \varphi(a_{j-1})P_{a_{j-1}} \subseteq L$. Thus $w_1 \cdots w_{j-1}w_j \in \varphi(a_1) \cdots \varphi(a_{j-1})P_{a_{j-1}}$, and now we have to apply the pumping lemma for the word w_j with respect to $P_{a_{j-1}}$. Then, $w_j = x_jy_jz_j$ with $|y_j| \geq 1$ and $x_jy_j^iz_j \in P_{a_{j-1}}$ for any $i \geq 1$. Consider $w' = w_1 \cdots w_j$, $x = w_1 \cdots w_{j-1}x_j$, $y = y_j$ and $z = z_j$ and the theorem is satisfied in this case too. □

4 Comparisons Between Families of Languages

The first remark of this section is that any family of L-type jumping Petri net languages is closed under "*" (the net jumps from any final marking to the initial marking). This fact also holds for the family $\mathbf{RP^f}$, $\mathbf{RP}$, $\mathbf{RP^\lambda}$, but not for $\mathbf{RP^f_{fin}}$, $\mathbf{RP_{fin}}$, $\mathbf{RP^\lambda_{fin}}$. The closure under "*" of the family $\mathbf{RP^\lambda_{fin}}$ can be proved using the following idea. At any reachable marking of the net some λ-transitions are enabled. These transitions will reset the current marking to the zero-marking $\mathbf{0}$ (all the components are 0) and then, a jump from $\mathbf{0}$ to the initial marking will restart the net. The non-closure under Kleene star of the families of Petri net languages leads us to the following results:

Theorem 4.1 *(1)* $\mathbf{P^\lambda} \subset \mathbf{RP^\lambda_{fin}}$;
(2) $\mathbf{L^f} \subset \mathbf{RL^f_{fin}}$, $\mathbf{L} \subset \mathbf{RL_{fin}}$, $\mathbf{L^\lambda} \subset \mathbf{RL^\lambda_{fin}}$.

Theorem 4.2 $\mathcal{L}_3 \subset \mathbf{RL^f_{fin}}$.

Proof The inclusion follows from the fact that $\mathcal{L}_3 = \mathbf{RL(fss)^f_{fin}} \subseteq \mathbf{RL^f_{fin}}$; it is strict because $\mathcal{L}_3 \cup \mathbf{L}^f \subseteq \mathbf{RL}^f_{fin}$ and the families $\mathcal{L}_3$ and $\mathbf{L}^f$ are incomparable ([11]). □

Theorem 4.3 $L = \{a^nb^n \mid n \geq 0\} \notin \mathbf{RL^f_{fin}}$.

Proof Suppose by contradiction that $L \in \mathbf{RL}^{\mathbf{f}}_{\mathbf{fin}}$. Then there exist a regular language L' and a substitution with λ-free languages $\varphi : alph(L') \longrightarrow \mathbf{L}^{\mathbf{f}}$ such that $L = \varphi(L')$.

Case 1 L' is infinite. There exist $u \in L'$ and a decomposition of u, $u = u_1u_2u_3$ such that $|u_2| \geq 1$ and $u_1u_2^iu_3 \in L'$ for any $i \geq 0$ (the pumping lemma for regular languages).

Since $\varphi(u_1u_2^iu_3) = \varphi(u_1)[\varphi(u_2)]^i\varphi(u_3) \subseteq L$ and $u_2 \neq \lambda$, it follows that there exist $w_1 \in \varphi(u_1), w_2 \in \varphi(u_2)$ and $w_3 \in \varphi(u_3)$ such that $w_1w_2^iw_3 \in L$ for any $i \geq 0$. It is easy to see that no matter how w_1, w_2, w_3 ($w_2 \neq \lambda$) are chosen we cannot have $w_1w_2^iw_3 \in L$ for any $i \geq 0$.

Case 2 L' is finite. If so, let $L' = \{u_1, \ldots, u_k\}$, $k \geq 1$. Since L is infinite, there exists $j \in \{1, \ldots, k\}$ such that $\varphi(u_j)$ is infinite. Let $u_j = a_1 \ldots a_{m_j}$, $m_j \geq 1$, and $\varphi(u_j) = \{a^{i_1}b^{i_1}, a^{i_2}b^{i_2}, \ldots\}$, where $0 \leq i_1 < i_2 < \ldots$. Then there is $i \in \{1, \ldots, m_j\}$ such that $\varphi(a_i)$ is infinite. We have to consider now the next cases.

If $\varphi(a_i) = \{a^{\alpha_1}, a^{\alpha_2}, \ldots\}$, where $0 \leq \alpha_1 < \alpha_1 < \ldots$, then it is easy to see that no matter how the words in $\varphi(a_i)$ are catenated to the left or to the right we cannot obtain only words in $\varphi(u_j)$. Similar reason tells that $\varphi(a_i)$ cannot be $\{b^{\beta_1}, b^{\beta_2}, \ldots\}$, where $0 \leq \beta_1 < \beta_2 < \ldots$.

As any subset of $\varphi(u_j)$ of cardinality at least two is not a member of $\mathbf{L}^{\mathbf{f}}$, the only case which remains to be considered is $\varphi(a_i) = \{a^{\alpha_1}b^{\beta_1}, a^{\alpha_2}b^{\beta_2}, \ldots\}$, where α's and β's are natural numbers and there is n such that $\alpha_n \neq \beta_n$. There is also an p, $p \neq n$, such that either $\alpha_n \neq \alpha_p$ or $\beta_n \neq \beta_p$. A straightforward analysis shows us that no matter how the language $\varphi(a_i)$ is catenated to the left or to the right we cannot obtain only words in $\varphi(u_j)$.

In both cases we have derived a contradiction and hence $L \notin \mathbf{RL}^{\mathbf{f}}_{\mathbf{fin}}$. □

Corollary 4.1 $\mathbf{RL}^{\mathbf{f}}_{\mathbf{fin}} \subset \mathbf{RL}_{\mathbf{fin}}$.

Proof $\{a^nb^n \mid n \geq 0\} \in \mathbf{L} \subset \mathbf{RL}_{\mathbf{fin}}$ and $\{a^nb^n \mid n \leq 0\} \notin \mathbf{RL}^{\mathbf{f}}_{\mathbf{fin}}$. □

Corollary 4.2 *The families* $\mathbf{RL}^{\mathbf{f}}_{\mathbf{fin}}$ *and* $\mathcal{L}_2$ *are incomparable.*

Proof $\{a^nb^n \mid n \geq 0\} \in \mathcal{L}_2 - \mathbf{RL}^{\mathbf{f}}_{\mathbf{fin}}$ and $\{a^ndb^nec^n \mid n \geq 1\} \in \mathbf{RL}^{\mathbf{f}}_{\mathbf{fin}} - \mathcal{L}_2$. □

5 Finite Jumping Nets and Global Inhibitor Nets

We will make a connection between finite jumping nets and a subclass of inhibitor nets, global inhibitor nets. We recall that ([9]) an inhibitor net is a pair $\gamma = (\Sigma, I)$, where Σ is a Petri net and $I \subseteq S \times T$ such that $I \cap F = \emptyset$.

In an inhibitor net γ the transition t is *i-enabled* at a marking M, abbreviated $M[t\rangle_{\gamma,i}$, iff $t^- \leq M$ and $M(s) = 0$ for any $s \in \{s \in S | (s,t) \in I\}$. If $M[t\rangle_{\gamma,i}$ then t may occur yielding a new marking M', abbreviated $M[t\rangle_{\gamma,i}M'$, given by $M' = M + \Delta t$. As we can see, an inhibitor net has the capability to perform zero-tests on some places. A *global inhibitor net* is defined as an inhibitor net performing zero-tests on all places, that is $(s,t) \in I \Rightarrow (s',t) \in I$, $\forall s' \in S$.

Now we show that $FJPTN$'s can be simulated by global inhibitor nets. Let $\gamma = (\Sigma, R, M_0, l)$ be an $FJPTN$ with only a jump, $R = \{(M, M')\}$. Construct the following inhibitor net (the net is picturially represented in Figure 6.1 and the relation I is given by $I = \{(s,t') | s \text{ is a place}\}$).

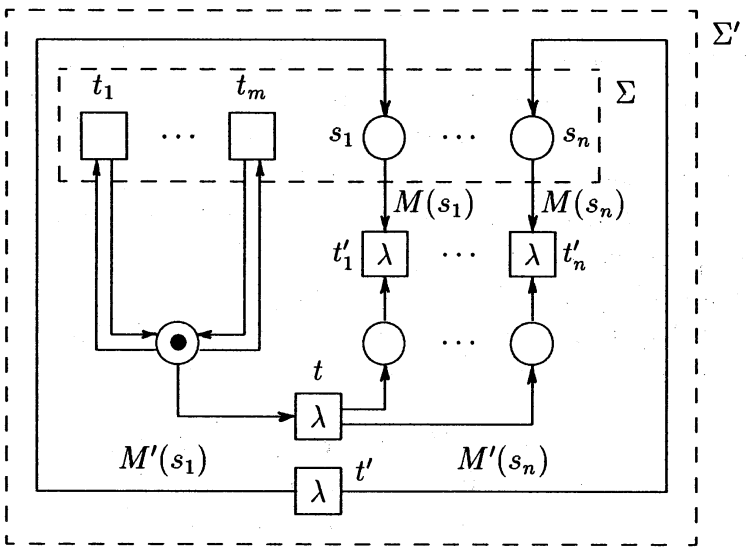

Figure 6.1

It is clear that t' performs a zero-test on all places and so this net is a global inhibitor net. Its activity can be described as follows:

- the transition t blocks Σ and then the transitions $t'_1, \ldots, t'_m$ check whether or not the current marking covers M (if all t'_i can occur than the current marking covers M). The zero-test performed by t' checks when the current marking is exactly M (t' can occur iff no token is in the net). If this is the case the marking M' is setted for Σ.

The above construction can be easily generalized to an $FJPTN$ with arbitrarily many jumps.

Now we show that any arbitrarily labelled global inhibitor net can be simulated by an $FJPTN$. Indeed, let $\gamma = (\Sigma, I, M_0, l)$ be such an inhibitor net. Assume $I = \{(s,t) | s \in S\}$, where t is a fixed transition. If $l(t) = \lambda$ then we can simulate the extent of change caused by the occurrence of t using the jump $(\mathbf{0}, M)$, where $M(s) = W(t,s)$ for all $s \in S$ (we recall that $I \cap F = \emptyset$, that is $W(s,t) = 0$ for all $s \in S$). If $l(t) = a \neq \lambda$ then we simulate the extent of change caused by t using the net Figure 6.2 and the jump $\{((\mathbf{0},1,0),(M,0,1))\}$ (M is as above). By this jump the net Σ will be blocked; it is relieved after occurring the transition t labelled by a (t being a new transition).

The generalization to an arbitrary global inhibitor net is straightforward.

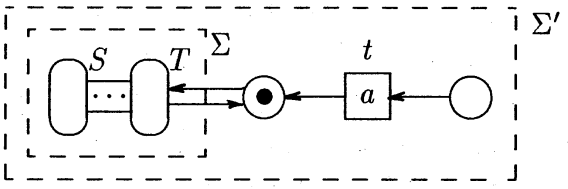

Figure 6.2

Final Remarks

We consider that the extension of Petri nets allowing finite jumps is quite reasonable: on the one hand such nets have the basic decision problems decidable and, on the other hand the finite jumps strictly increase the power of the nets.

We close with some important open problems, in our estimation.

P1. Are the inclusions $\mathbf{RL_{fin}} \subseteq \mathcal{L}_1$, $\mathbf{RL^{\lambda}_{fin}} \subseteq \mathcal{L}_{rec}$, $\mathbf{RL_{fin}} \subseteq \mathbf{RL^{\lambda}_{fin}}$, $\mathbf{RP_{fin}} \subseteq \mathbf{RL_{fin}}$, $\mathbf{RL^{\lambda}_{fin}} \subseteq \mathbf{RL^{\lambda}_{fin}}$ proper or not?

P2. Are the families $\mathbf{RP^{f}_{fin}}$ and $\mathbf{RP_{fin}}$ closed under "*"?

P3. Define $\mathbf{RX^{f}_{k}}$ ($\mathbf{RX_{k}}$, $\mathbf{RX^{\lambda}_{k}}$, resp.) as being the family of X-type languages generated by jumping nets having at most k jumps ($k \geq 0$), that is $|R| \leq k$. We have

$$\begin{array}{ccccccc} \mathbf{X^{f}} & \subseteq & \mathbf{RX^{f}_{k}} & \subseteq & \mathbf{RX^{f}_{k+1}} & \subseteq & \mathbf{RX^{f}_{fin}} \\ \mathbf{X} & \subseteq & \mathbf{RX_{k}} & \subseteq & \mathbf{RX_{k+1}} & \subseteq & \mathbf{RX_{fin}} \\ \mathbf{X^{\lambda}} & \subseteq & \mathbf{RX^{\lambda}_{k}} & \subseteq & \mathbf{RX^{\lambda}_{k+1}} & \subseteq & \mathbf{RX^{\lambda}_{fin}} \end{array}$$

for all $k \geq 1$ and $X \in \{P, L\}$.

Does this restriction define proper hierarchies of jumping Petri net languages?

P4. What about the connection between $FJPTN$'s and global inhibitor nets in the case we do not allow λ-transitions?

References

[1] E. Best, C. Fernandez: *Notations and Terminology on Petri Net Theory*, Arbeitspapiere der GMD 195, 1986.

[2] H.D. Burkhard: *On priorities of parallelism; Petri nets under the maximum firing strategy*, Int. Conf. "LOGLAN 77", Poznan, 1980.

[3] H.D. Burkhard: *The Maximum Firing Strategy in Petri Nets Gives More Power*, ICS-PAS Report 441, Warszaw, 24-2, 26, 1980.

[4] H.D. Burkhard: *Two pumping lemmata for Petri nets*, EIK, vol. 17, no. 7, 1981, 349 - 362.

[5] H.D. Burkhard: *Ordered firing in Petri nets*, EIK, vol. 17, no. 2/3, 1981, 71 - 86.

[6] H.D. Burkhard: *Control of Petri Nets by Finite Automata*, Preprint 26, Sektion Mathematik, Humboldt-Universität, Berlin, 1982.

[7] H.D. Burkhard: *What Gives Petri Nets More Computational Power*, Preprint 45, Sektion Mathematik, Humboldt-Universität, Berlin, 1982.

[8] H.J.M. Goeman, L.P.J. Groenwegen, H.C.M. Kleijn, G.Rozenberg: *Constrained Petri nets (Part I, II)*, Fundamenta Informaticae, vol. 6, no. 1, 1983.

[9] M. Hack: *Petri Net Languages*, CSG Memo 124, Project MAC, MIT, 1975.

[10] J.E. Hopcroft, J.D. Ullman: *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, Mass., 1979.

[11] M. Jantzen: *Language Theory of Petri Nets*, LNCS 254, Springer-Verlag, 1986.

[12] T. Jucan, C. Masalagiu, F.L. Ţiplea: *Relation Based Controlled Petri Nets*, Scientif Annals of the "Al. I. Cuza" University, Informatics Section, Tom 2, 1995.

[13] W. Reisig: *Petri Nets. An Introduction*, EATCS Monographs on Theoret. Comput. Sci., Springer-Verlag, 1985.

[14] W. Reisig: *Place Transition Systems*, LNCS 254, Springer-Verlag, Berlin, Heidelberg, 1986.

[15] G. Rozenberg, R. Verraedt: *Restricting the in-out structure of graph of Petri nets*, Fundamenta Informaticae, vol. 7, no. 2, 1984.

[16] F.L. Ţiplea, T. Jucan, C. Masalagiu: *Conditional Petri net languages*, J. Inf. Process. Cybern. EIK, vol. 27, no. 1, 1991, 55 - 66.

[17] F.L. Ţiplea: *Selective Petri net languages*, Intern. J. Computer Math., vol. 43, no. 1-2, 1992, 61 - 80.

[18] F.L. Ţiplea, T. Jucan: *Jumping Petri Nets*, Foundations of Computing and Decision Sciences, vol. 19, no. 4, 1994, 319 - 332.

[19] F.L. Ţiplea: *On Conditional Grammars and Conditional Petri Nets*, in: Mathematical Aspects of Natural and Formal Languages (Gh. Păun, ed.), World Scientific, Singapore, 1995, 431 - 455.

[20] T. Ushio: *On controllability of controlled Petri nets*, Control Theory and Advanced Technology, vol. 5, no. 3, 1989, 265 - 277.

[21] R. Valk: *Self-modifying nets, a natural extension of Petri nets*, LNCS 62, Springer-Verlag, 1978.

[22] R. Valk: *On the computational power of extended Petri nets*, LNCS 64, Springer-Verlag, 1978.